Welcome To:
Module 4

Linux Fundamentals

Commands Syntax

- Command options and arguments

Commands typically have the syntax:

command option(s) argument(s)

Options:

Modify the way that a command works

Usually consist of a hyphen or dash followed by a single letter

Some commands accept multiple options which can usually be grouped together after a single hyphen

Arguments:

Most commands are used together with one or more arguments

Some commands assume a default argument if none is supplied

Arguments are optional for some commands and required by others

File Permissions

- UNIX is a multi-user system. Every file and directory in your account can be protected from or made accessible to other users by changing its access permissions. Every user has responsibility for controlling access to their files.

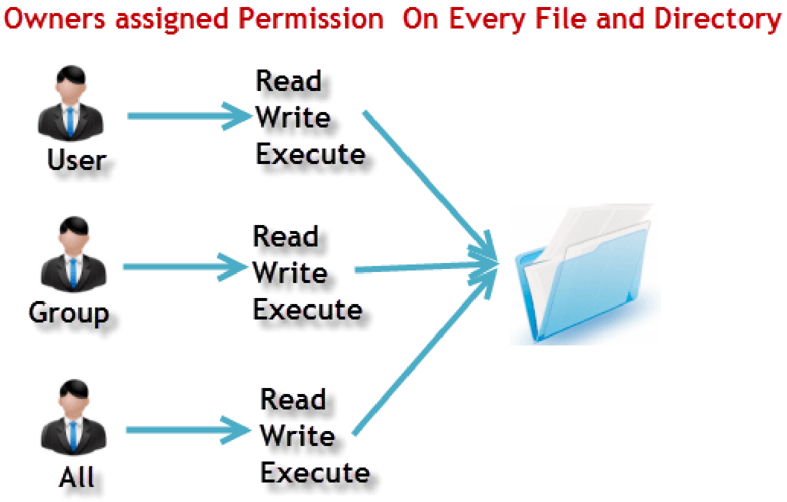
- Permissions for a file or directory may be restricted to by types
- There are 3 type of permissions
  - r - read
  - w - write
  - x - exeawke = running a program
- Each permission (rwx) can be controlled at three levels:
  - u - user = yourself
  - g - group = can be people in the same project
  - o - other = everyone on the system
- File or Directory permission can be displayed by running ls –l command
  - -rwxrwxrwx
- Command to change permission
  - chmod

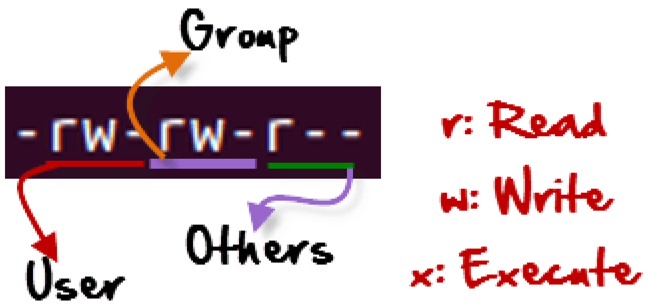
Permission Using Numeric Mode

- Permission to a file and directory can also be assigned numerically
  - `chmod ugo+r FILE`

  OR
  - `chmod 444 FILE`

```
-r--r--r--
```



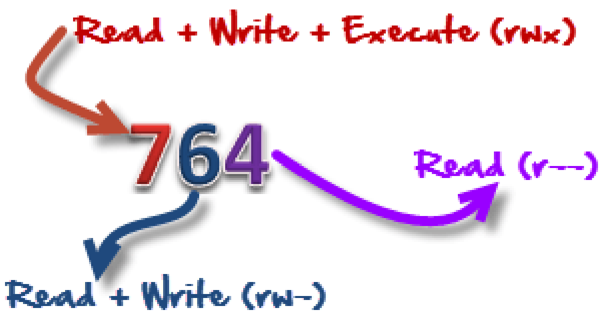


Permission Using Numeric Mode

- The table below assigns numbers to permissions types

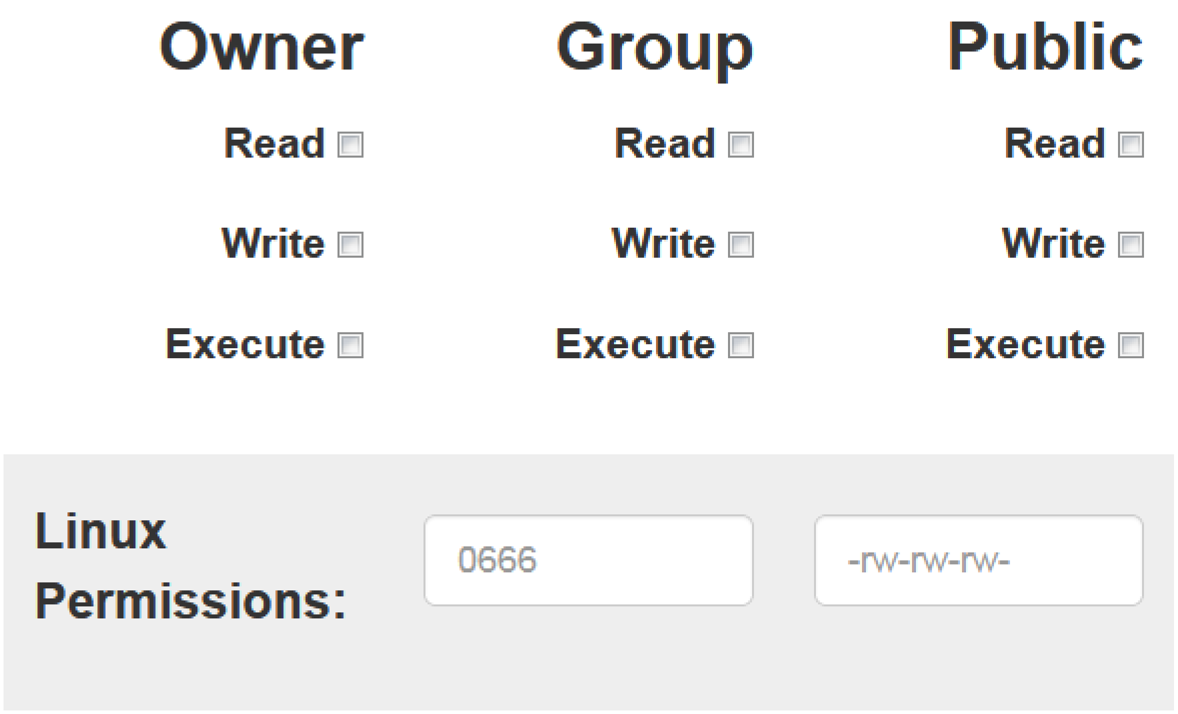
Number	Permission Type	Symbol
0	No Permission	---
1	Execute	--x
2	Write	-w-
3	Execute + Write	-wx
4	Read	r--
5	Read + Execute	r-x
6	Read +Write	rw-
7	Read + Write +Execute	rwx

- chmod 764 FILE



Permission Using Numeric Mode

- Online calculators can be used as well



File Ownership

- There are 2 owners of a file or directory
  - User and group

- Command to change file ownership
  - chown and chgrp
    - chown changes the ownership of a file
    - chgrp changes the group ownership of a file

- Recursive ownership change option (Cascade)
  - -R

Help Commands

- There are 3 types of help commands
  - **`whatis`** `command`
  - `command` **`--help`**
  - **`man`** `command`

TAB Completion and Up Arrow

- Hitting TAB key completes the available commands, files or directories
  - **chm TAB**
  - **ls j<TAB>**
  - **cd Des<TAB>**

- Hitting up arrow key on the keyboard returns the last command ran.

Adding Text to Files (Redirects)

- 3 Simple ways to add text to a file
    - **`vi`**
    - **`Redirect command output > or >>`**
    - **`echo > or >>`**

Input and Output Redirects

- There are 3 redirects in Linux
  1. Standard input (**`stdin`**) and it has file descriptor number as 0
  2. Standard output (**`stdout`**) and it has file descriptor number as 1
  3. Standard error (**`stderr`**) and it has file descriptor number as 2

- Output (**stdout**) - 1
  - By default when running a command its output goes to the terminal
  - The output of a command can be routed to a file using > symbol
    - E.g.

```
ls -l > listings
pwd > findpath
```

  - If using the same file for additional output or to append to the same file then use >>
    - E.g.

```
ls -la >> listings
echo "Hello World"  >> findpath.
```

Input and Output Redirects

- Input (**stdin**) - 0
  - Input is used when feeding file contents to a file
    - E.g.
      ```
      cat < listings
      mail –s "Office memo" allusers@abc.com < memoletter
      ```

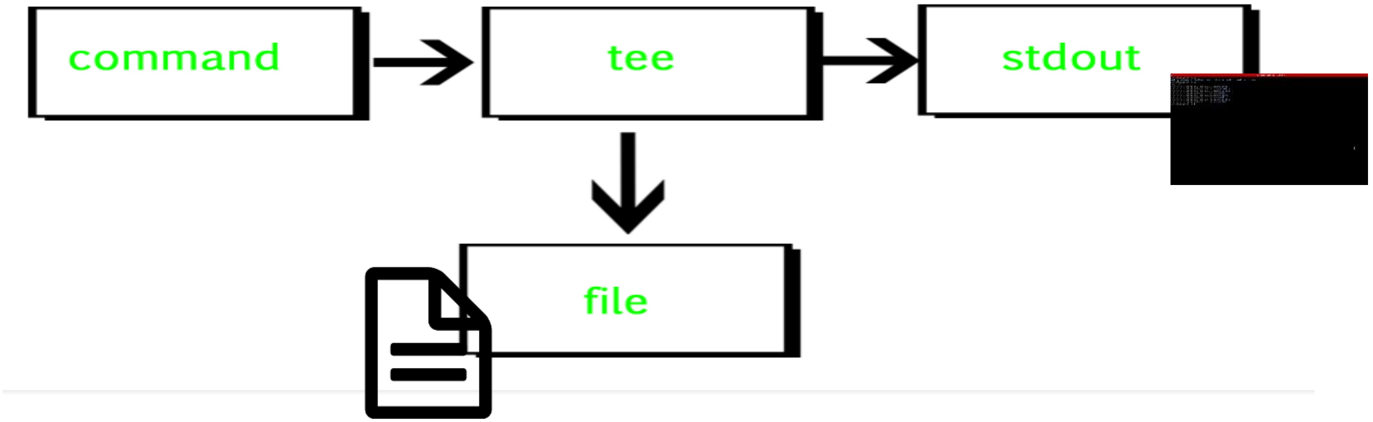
- Error (**stderr)** - 2
  - When a command is executed we use a keyboard and that is also considered (stdin -0)
  - That command output goes on the monitor and that output is (stdout – 1)
  - If the command produced any error on the screen then it is considered (stderr – 2)
    - We can use redirects to route errors from the screen
      - E.g
        ```
        ls –l /root 2> errorfile
        telnet localhost 2> errorfile.
        ```

Standard Output to a File (tee)

- “tee” command is used to store and view (both at the same time) the output of any command

- The command is named after the T-splitter used in plumbing. It basically breaks the output of a program so that it can be both displayed and saved in a file. It does both the tasks simultaneously, copies the result into the specified files or variables and also display the result.





Pipes

- A pipe is used by the shell to connect the output of one command directly to the input of another command.

The symbol for a pipe is the vertical bar (|). The command syntax is:

```
command1 [arguments]  |  command2 [arguments]
```





Pipes



`ls -l` `|` `more`

File Maintenance Commands

- `cp`
- `rm`
- `mv`
- `mkdir`
- `rmdir` or `rm -r`
- `chgrp`
- `chown`

File Display Commands

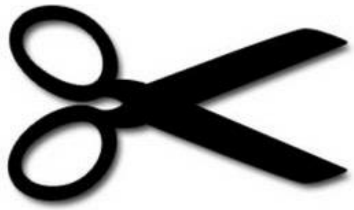
- `cat`
- `more`
- `less`
- `head`
- `tail`

Filters / Text Processors Commands

- `cut`
- `awk`
- `grep and egrep`
- `sort`
- `uniq`
- `wc`

`cut` - Text Processors Commands

`cut`



- cut is a command line utility that allows you to cut parts of lines from specified files or piped data and print the result to standard output. It can be used to cut parts of a line by delimiter, byte position, and character

• `cut filename`	=	Does not work
• `cut --version`	=	Check version
• `cut –c1 filename`	=	List one character
• `cut –c1,2,4`	=	Pick and chose character
• `cut –c1-3 filename`	=	List range of characters
• `cut –c1-3,6-8 filename`	=	List specific range of characters
• `cut –b1-3 filename`	=	List by byte size
• `cut -d: -f 6 /etc/passwd`	=	List first 6th column separated by :
• `cut -d: -f 6-7 /etc/passwd`	=	List first 6 and 7th column separated by :
• `ls –l	cut –c2-4`	=	Only print user permissions of files/dir

awk - Text Processors Commands

awk



- awk is a utility/language designed for data extraction. Most of the time it is used to extract fields from a file or from an output

- **awk --version** = Check version
- **awk ‘{print $1}’ file** = List 1st field from a file
- **ls –l | awk ‘{print $1,$3}’** = List 1 and 3rd field of ls –l output
- **ls –l | awk ‘{print $NF}’** = Last field of the output
- **awk '/Jerry/ {print}' file** = Search for a specific word
- **awk -F: '{print $1}' /etc/passwd** = Ouput only 1st field of /etc/passwd
- **echo "Hello Tom" | awk '{$2="Adam"; print $0}`** = Replace words field words
- **cat file | awk '{$2=“Imran"; print $0}`** = Replace words field words
- **awk 'length($0) > 15` file** = Get lines that have more than 15 byte size
- **ls -l | awk '{if($9 == "seinfeld") print $0;}`** = Get the field matching seinfeld in /home/iafzal
- **ls -l | awk '{print NF}`** = Number of fields.

grep/egrep - Text Processors Commands



- What is grep?
    - The grep command which stands for "global regular expression print," processes text line by line and prints any lines which match a specified pattern

- **grep --version OR grep --help** = Check version or help
- **grep keyword file** = Search for a keyword from a file
- **grep -c keyword file** = Search for a keyword and count
- **grep -i KEYword file** = Search for a keyword ignore case-sensitive
- **grep -n keyword file** = Display the matched lines and their line numbers
- **grep -v keyword file** = Display everything but keyword
- **grep keyword file | awk '{print $1}'** = Search for a keyword and then only give the 1st field
- **ls -l | grep Desktop** = Search for a keyword and then only give the 1st field

- **egrep -i "keyword|keyword2" file** = Search for 2 keywords.

sort/uniq - Text Processors Commands

- What are sort and uniq commands?
    - Sort command sorts in alphabetical order
    - Uniq command filters out the repeated or duplicate lines

- **`sort --version OR sort --help`** = Check version or help
- **`sort file`** = Sorts file in alphabetical order
- **`sort -r file`** = Sort in reverse alphabetical order
- **`sort -k2 file`** = Sort by field number

- **`uniq file`** = Removes duplicates
- **`sort file | uniq`** = Always sort first before using uniq their line numbers
- **`sort file | uniq -c`** = Sort first then uniq and list count
- **`sort file | uniq -d`** = Only show repeated lines.

wc - Text Processors Commands



- What is **wc** command?
  - The command reads either standard input or a list of files and generates: **newline count, word count, and byte count**

- **`wc --version OR wc --help`** = Check version or help
- **`wc file`** = Check file line count, word count and byte count
- **`wc –l file`** = Get the number of lines in a file
- **`wc –w file`** = Get the number of words in a file
- **`wc –b file`** = Get the number of bytes in a file
- **`wc DIRECTORY`** = NOT allowed
- **`ls –l | wc -l`** = Number of files
- **`grep keyword | wc -l`** = Number of keyword lines.

Compare Files

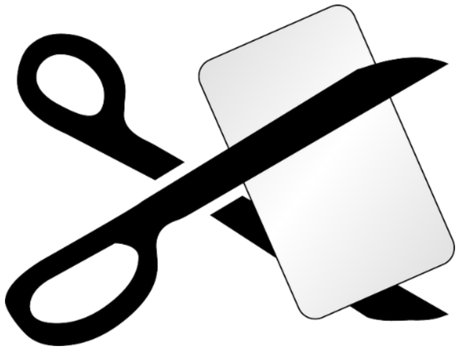
- **diff** (Line by line)
- **cmp** (Byte by byte)

Compress and un-Compress Files

- `tar`
- `gzip`
- `gzip -d OR gunzip`

Truncate File Size (truncate)

- The Linux **`truncate`** command is often used to shrink or extend the size of a file to the specified size



- Command
  - **`truncate –s 10 filename`**

Combining and Splitting Files

- Multiple files can be combined into one and
- One file can be split into multiple files

  - `cat file1 file2 file3 > file4`
  - `split file4`

  - `e.g. split -l 300 file.txt childfile`

  Split file.txt into 300 lines per file and output to childfileaa, childfileab and childfileac

Linux vs. Windows Commands

Command Description	Windows	Linux
Listing of a directory	dir	ls -l
Rename a file	ren	mv
Copy a file	copy	cp
Move file	move	mv
Clear screen	**cls**	clear
Delete file	del	rm
Compare contents of files	fc	diff
Search for a word/string in a file	find	grep
Display command help	**command /?**	man command
Displays your location in the file system	**chdir**	pwd
Displays the time	**time**	date